Artificial intelligence lab

Sumalata
1BM19CS409
IInd IA

1)

```
import re
def getAttributes(expression):
    expression = expression.split("(")[1:]
    expression = "(".join(expression)
    expression = expression.split(")")[:-1]
    expression = ")".join(expression)
    attributes = expression.split(",")
    return attributes

def getInitialPredicate(expression):
    return expression.split("(")[0]

def isConstant(char):
    return char.isupper() & len(char)==1

def isVariable(char):
    return char.islower() & len(char)==1

def replaceAttributes(exp, old, new):
    attributes = getAttributes(exp)
    Predicate = getInitialPredicate(exp)
    for index, val in enumerate(attributes):
        if val == old:
            attributes[index] = new
    return Predicate + "(" + ",".join(attributes) +
```

AI lab

Sumalata
1BM19CS409

```
def apply(exp, substitutions):
    for substitution in substitutions:
        new, old = substitution
        exp = replaceAttributes(exp, old, new)
    return exp

def checkOccurs(var, exp):
    if exp.find(var) == -1:
        return False
    return True

def getFirstPart(expression):
    attributes = getAttributes(expression)
    attributes = getAttributes(expression)
    newExpression = Predicate + "(" + ",".join
                    (attributes[1:]) + ")"
    return newExpression

def unify(exp1, exp2):
    if exp1 == exp2:
        return []
```

```
if    isConstant(exp1) & isConstant(exp2):
    if exp1 != exp2:
        Print(f"{exp1} & {exp2} are constants.
            cannot be unified")
        return []
    if isConstant(exp1):
        return [(exp1, exp2)]
    if isConstant(exp2):
        return [(exp2, exp1)]
    if isVariable(exp1):
        return [(exp2, exp1)] if not checkOccurs
            (exp1, exp2) else []
    if isVariable(exp2):
        return ((exp1, exp2)] if not checkOccurs
            (exp2, exp1) else []
    if getInitialPredicate(exp1) != getInitialPredicate
            (exp2):
```

```
if initalSubstitution != []:
    tail1 = apply(tail1, initalSubstitution)
    tail2 = apply(tail2, initalSubstitution)

remainingSubstitution + remainingSubstitution
if not remainingSubstitution:
    return()

def main():
    Print("enter the first expression")
    e1 = input()
    Print("enter the second expression")
    e2 = input()
    substitutions = unify(e1, e2)
    Print("The Substitutions are:")
    Print('['/'.join(substitution) for
        substitution in substitutions])

main()
```